**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра Систем автоматизированного проектирования**

**отчет**

**по лабораторной работе №9**

**по дисциплине «Базы данных»**

**Тема: Создание UDF**

|  |  |  |
| --- | --- | --- |
| Студенты гр. 2308 |  | Попов Н.А. |
|  |  | Бебия Р.А. |
|  |  | Чиков А.А. |
| Преподаватель |  | Горяинов С.В. |

Санкт-Петербург

2024

**Цель работы**

Цель работы заключается в том, чтобы научиться писать и применять функции, определяемые пользователем (UDF). В лабораторной работе используется база данных AdventureWorks.

**Выполнение работы**

**Упражнение 1 – создание скалярной функции**

Запрос 1: Создание определяемой пользователем скалярной функции Sales.GetMaximumDiscountForCategory, которая находит максимальный процент скидки (поле DiscountPct), доступный на данный момент для конкретной категории. При этом функция имеет параметр @Category nvarchar(50) для ограничения результатов на основе категории и использует функцию GETDATE() для ограничения строк на основе доступности скидки на данный момент в диапазоне StartDate и EndDate.

CREATE FUNCTION Sales.GetMaximumDiscountForCategory (@Category varchar(50))

RETURNS decimal(18,2)

AS

BEGIN

DECLARE @MaxDiscount decimal(18,2);

SELECT @MaxDiscount = MAX(DiscountPct)

FROM Sales.SpecialOffer

WHERE GETDATE() BETWEEN StartDate AND EndDate

AND Category = @Category

RETURN @MaxDiscount;

END;

GO

Результат выполнения запроса представлен на Рисунке 1.

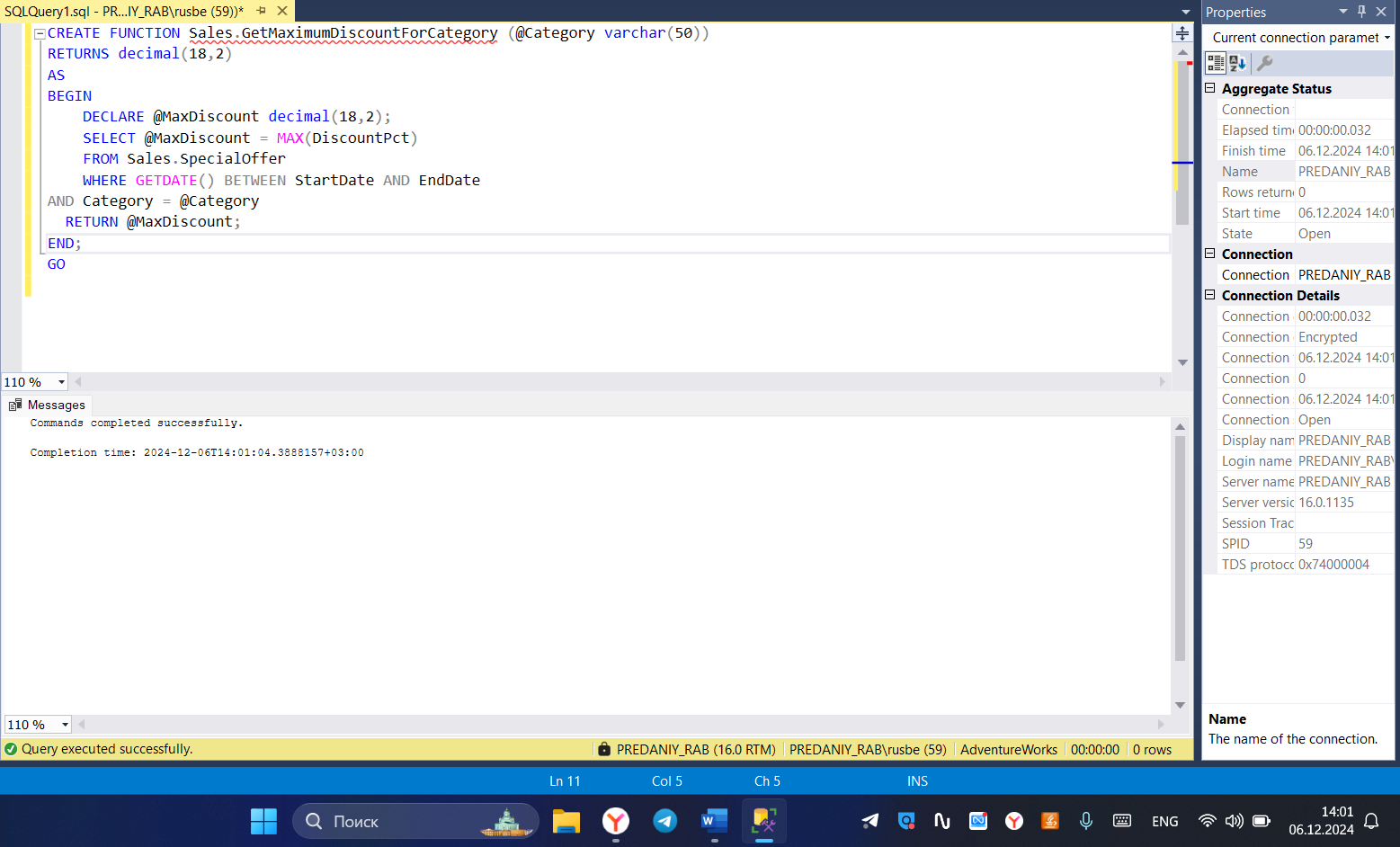


Рисунок 1 – Результат выполнения запроса

Запрос 2: Проверка функции.

SELECT Sales.GetMaximumDiscountForCategory('Reseller')

Результат выполнения запроса представлен на Рисунке 2.

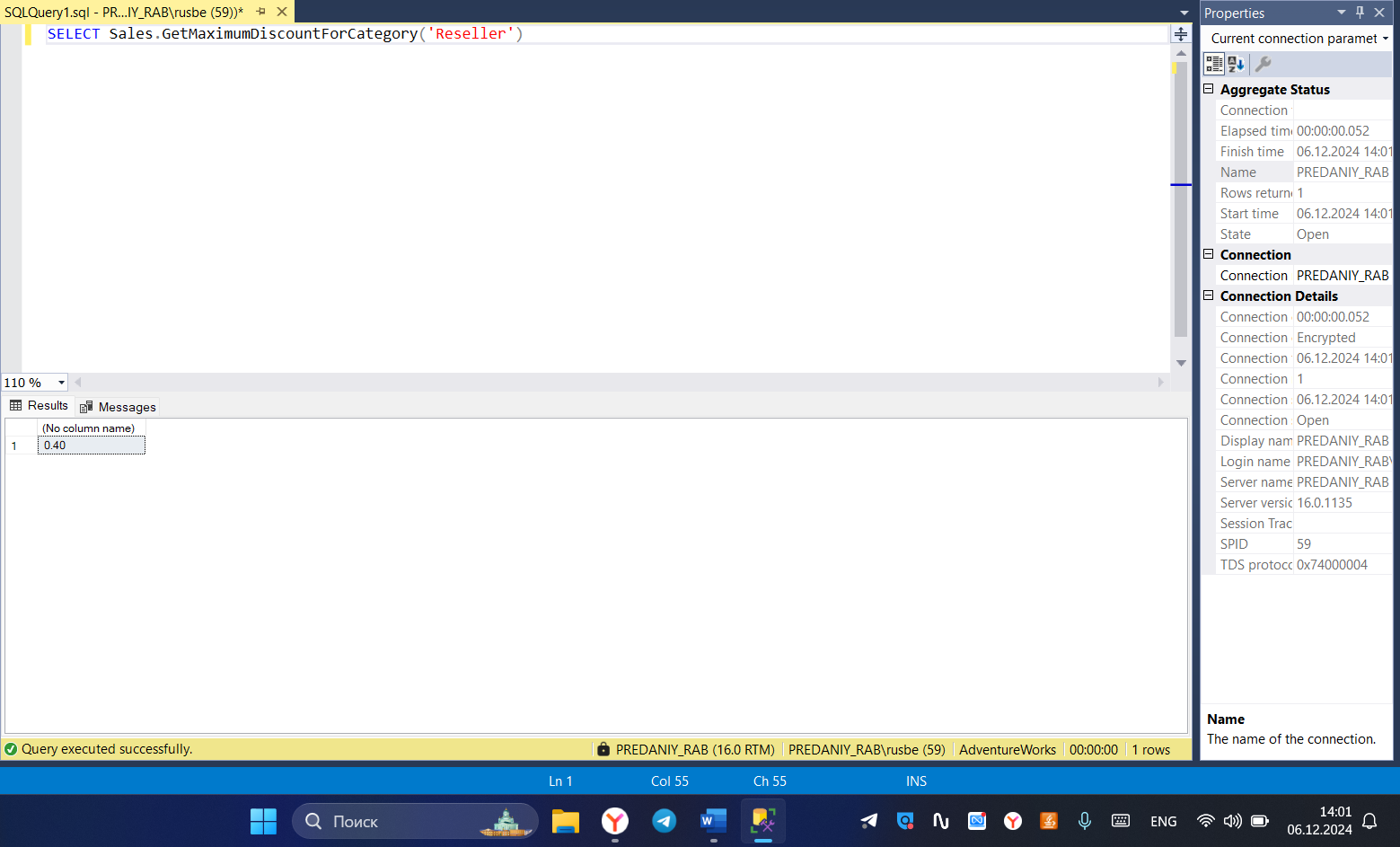


Рисунок 2 – Проверка функции

**Упражнение 2 – создание функции, возвращающей табличное значение (In-Line Table-valued UDF)**

Запрос 1: Создание функции Sales.GetDiscountsForDate, которая находит те же столбцы, что и хранимая процедура GetDiscounts (лабораторная работа 8, упражнение 1). У функции имеется входной параметр @DateToCheck datetime, используемый для фильтрации скидок на основе введенной даты.

CREATE FUNCTION Sales.GetDiscountsForDate (@DateToCheck datetime)

RETURNS TABLE

AS

RETURN SELECT Description, DiscountPct, Category, StartDate, EndDate, MinQty, MaxQty

FROM Sales.SpecialOffer

WHERE @DateToCheck BETWEEN StartDate AND EndDate

);

GO

Результат выполнения запроса представлен на Рисунке 3.

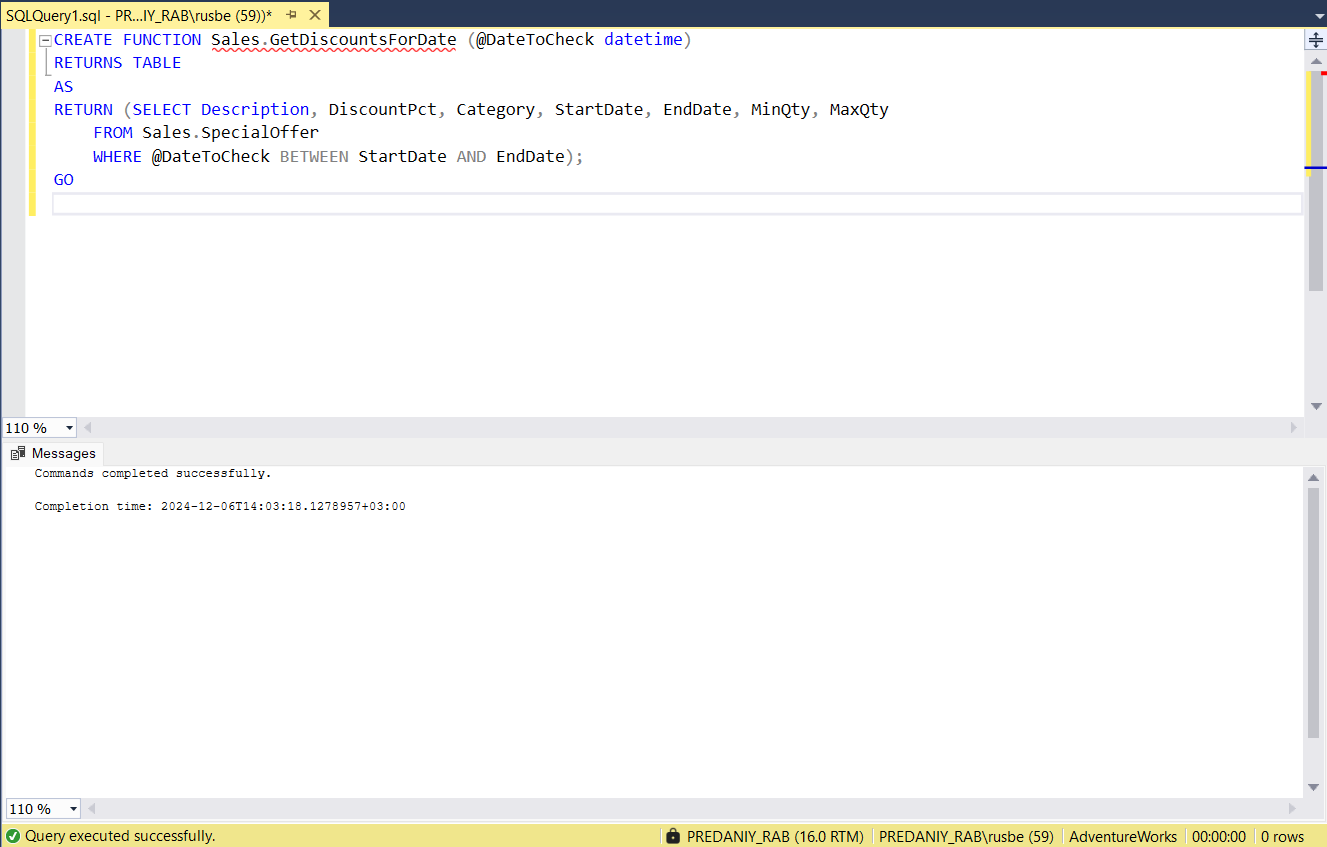


Рисунок 3 – Результат выполнения запроса

Запрос 2: Проверка функции.

SELECT \*

FROM Sales.GetDiscountsForDate(GetDate())

ORDER BY DiscountPct DESC

Результат выполнения запроса представлен на Рисунке 4.

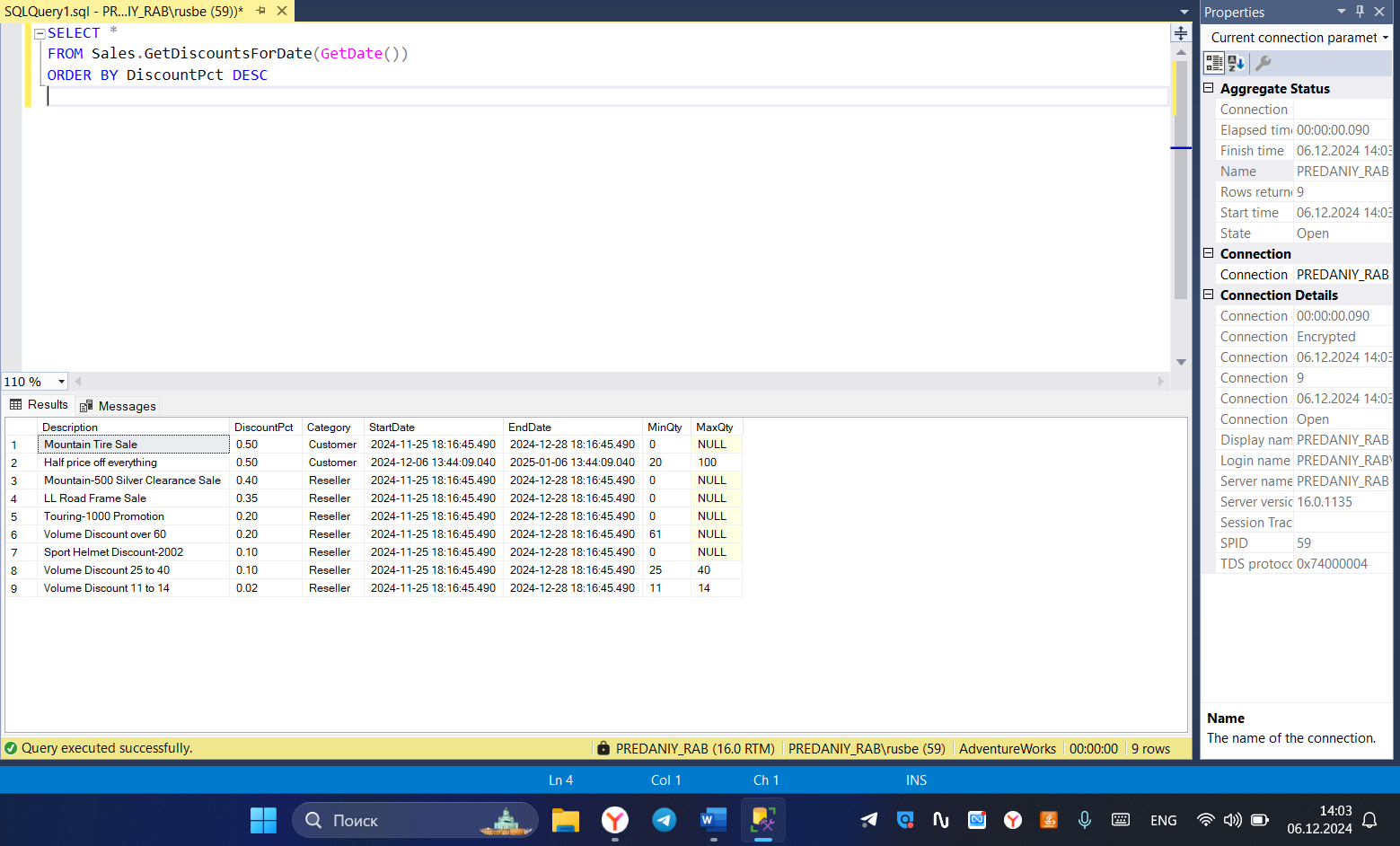


Рисунок 4 – Проверка функции

**Упражнение 3 – создание функции, возвращающей табличное значение (Multi-Statement Table-valued UDF)**

Запрос 1: Создание функции GetDiscountedProducts в схеме Sales. В этой функции написан запрос для поиска продуктов, имеющих скидку. Для формирования запроса соединены следующие таблицы: Sales.SpecialOfferProduct, Sales.SpecialOffer и Production.Product.

Запрос выводит следующие данные: столбцы ProductID, Name, ListPrice из таблицы Production.Product, столбцы Description и DiscountPct из таблицы Sales.SpecialOffer, а также два вычисляемых столбца. Первый вычисляемый столбец получается в результате произведения значений из поля ListPrice на DiscountPct; второй – в результате вычитания из ListPrice произведения значений поля ListPrice на DiscountPct. У функции имеется параметр @IncludeHistory bit, который применяется для фильтрации возвращенной таблицы на основе того, требуются ли сведения об истории скидок или необходимы только текущие сведения.

CREATE FUNCTION Sales.GetDiscountedProducts (@IncludeHistory bit)

RETURNS TABLE

AS

RETURN

(

SELECT p.ProductID, p.Name, p.ListPrice, so.Description AS DiscountDescription,

so.DiscountPct AS DiscountPercentage, p.ListPrice \* so.DiscountPct AS DiscountAmount,

p.ListPrice - (p.ListPrice \* so.DiscountPct) AS DiscountedPrice

FROM Production.Product p

JOIN Sales.SpecialOfferProduct sop ON p.ProductID = sop.Product1D

JOIN Sales.SpecialOffer so ON sop.SpecialOfferID = so.Special0fferID

WHERE (@IncludeHistory = 1 OR so.EndDate >= GetDate())

)

END;

GO

Результат выполнения запроса представлен на Рисунке 5.

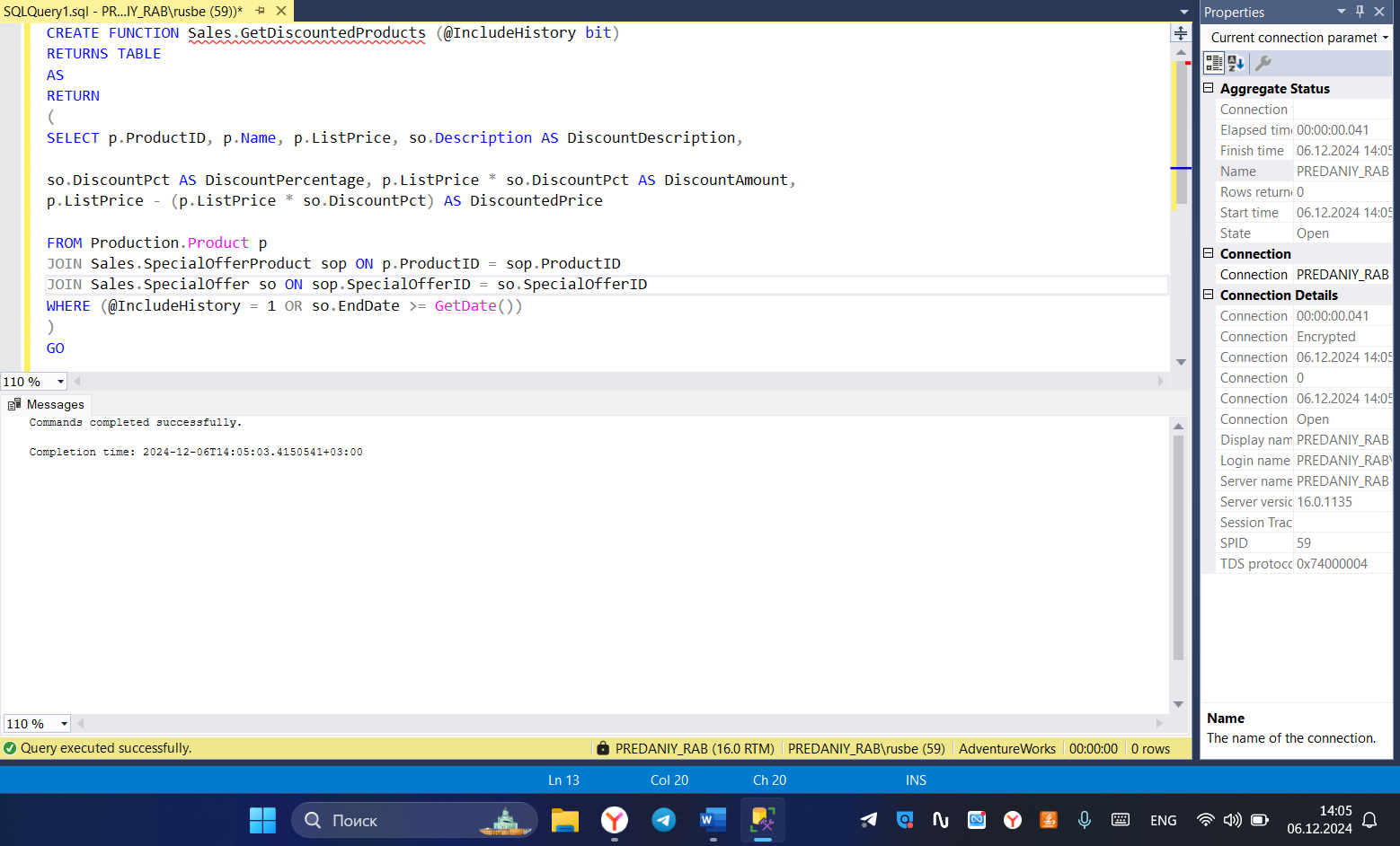


Рисунок 5 – Результат выполнения запроса

Запрос 2: Проверка функции.

SELECT \* FROM Sales.GetDiscountedProducts(0)

SELECT \* FROM Sales.GetDiscountedProducts(1)

Результат выполнения запроса представлен на Рисунке 6.

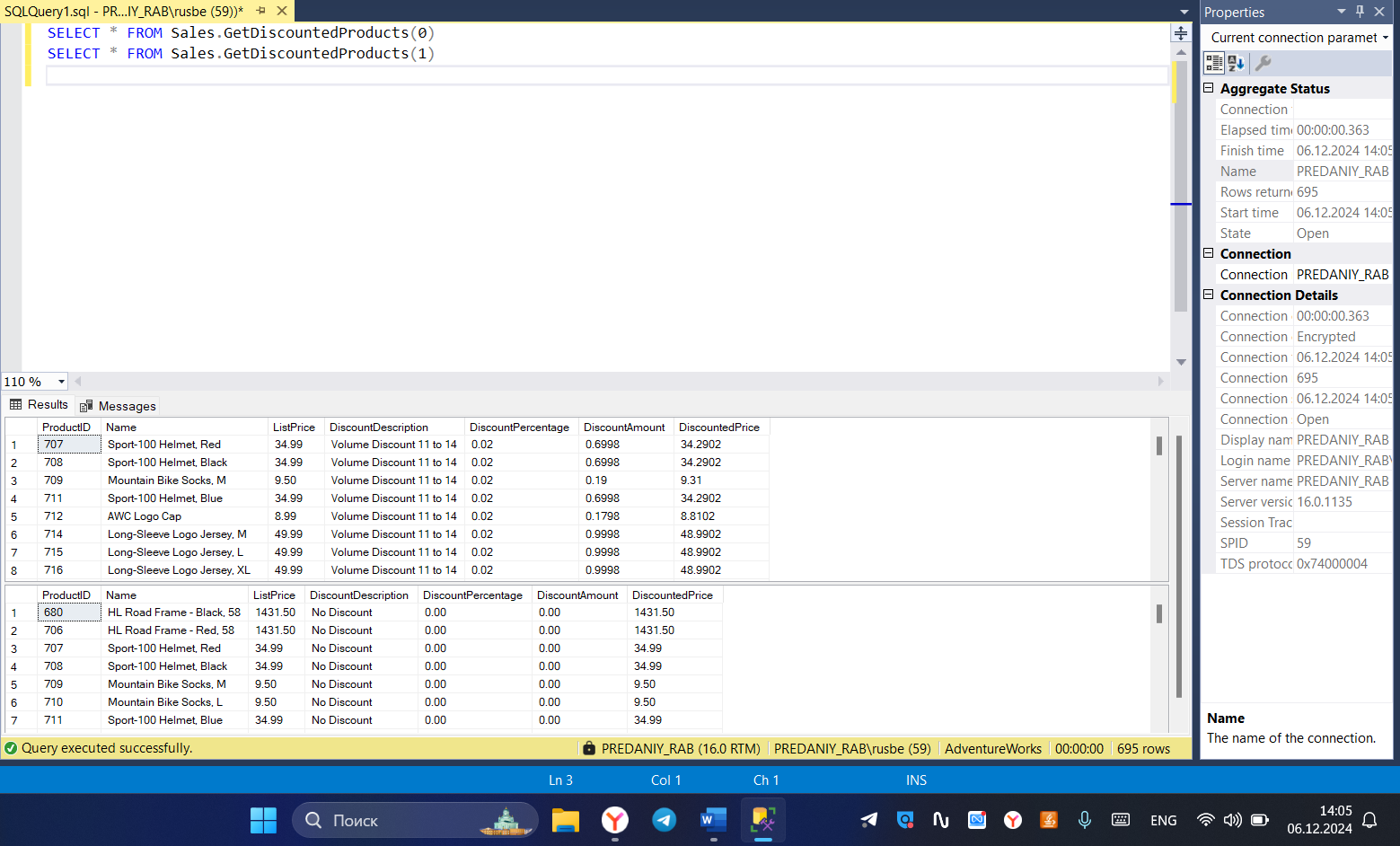


Рисунок 6 – Проверка функции

**Вывод**

В ходе лабораторной работы были изучены определяемые пользователем функции (UDF), включая скалярные функции и функции, возвращающие табличные значения, такие как In-Line Table-valued UDF и Multi-Statement Table-valued UDF. Скалярные функции позволяют выполнять вычисления и возвращать одно значение, что упрощает обработку данных. В свою очередь, функции типа In-Line Table-valued UDF предназначены для извлечения наборов данных с учетом заданных параметров, что значительно повышает эффективность работы с информацией. Функции Multi-Statement Table-valued UDF предлагают возможность реализовать более сложную логику извлечения данных благодаря использованию нескольких операторов внутри функции.Применение этих функций способствует более эффективному управлению данными и упрощает процесс разработки, позволяя разработчикам сосредоточиться на бизнес-логике и улучшении производительности запросов.